"linear inequalities"

This animation demonstrates, step-by-step, how to graph linear inequalities of the form y = mx + b and allows any two inequalities to be graphed simultaneously to find their intersection (solution to the system).

How to run this animation:

1. To set new values for m and b, double-click the pink value. A window opens. Type value. Press "OK".
2. Press one of the 4 pink buttons (the one that corresponds to the inequality desired). Several events happen with pauses in between: the specific inequality appears at the top of the screen, the y-intercept is highlighted, other border points are found using slope, the border line appears as dashed or solid, arrows show which way to shade along with a text explanation, one side is shaded, and a test point (with data) appears.
3. Drag the test point to verify the shading. "True" or "False" appears depending on whether or not the test point's ordered pair solves the inequality. There also is a button to show/hide instructions explaining how to "glue" the test point to the border, and remove it.
4. To overlay a second inequality graph, press the blue "Graph 2nd Ineq?" button and proceed just as had been done for the pink graph.
5. At any time, press the "Reset" button to restore the screen to its initial state (values of m & b do not change).

"parallel & perpendicular"

This animation explores the special slope relationships of parallel lines and perpendicular lines.

How to run this animation:

1. Press the "Create Parallel" button. A moving parallel line separates from $\overleftrightarrow{AB}$. Drag A, B, and line h.
2. Press the "Create Perpendicular" button. A moving line rotates out of $\overleftrightarrow{AB}$ and becomes perpendicular to it. There is a lot of dynamic data to watch here. Press the button a few times or use the Motion Controller (from the top pull-down menu "Display", release on "Show Motion Controller", drag to convenient place on screen) to pause the action to see all the *dynamic* (changing) data during the action: rotation angle, coordinates of Q becoming (-y,x) as compared to P (x,y), the slope of the rotating line, and product of the slope of $\overleftrightarrow{AB}$ and the slope of the rotating line.
3. Drag A, B, and line k. The special relationship, (x,y) and (-y,x), between P and its rotated image, Q, is relative to their intersection point. Hence it shows up clearly whenever the intersection point is at the origin.
4. At any time, press the "Reset" button to restore the screen to its initial state.

"mdpt formula"

Using any segment, this animation develops the formula for its midpoint, step-by-step.

How to run this animation:

1. Press button #1. $\overline{AB}$ is broken into its vertical and horizontal components and their intersection is graphed.
2. Press button #2. The vertical component's midpoint is calculated using the average of the y-coordinates.
3. Press button #3. The horizontal component's midpoint is calculated using the average of the x-coordinates.
4. Press button #4. These two midpoints merge to create the ordered pair of the midpoint of $\overline{AB}$. The formula appears at the top of the screen.
5. One at a time, drag points A and B. The *dynamic* ordered pairs in the top right corner (along with all the other data) adjust for the movement.
6. If you want points A and B to have integer coordinates only, go to the top pull-down menu "Graph" and release on "Snap Points". When dragged, they will jump to these spots.
7. At any time, press the "Reset" button to restore the screen to its initial state.

"dist formula"

Using any segment, this animation develops the formula for its length, step-by-step.

How to run this animation: Stress that Δx means "change in x" (coordinates), and Δy means "change in y".

1. Press button #1. $\overline{AB}$ is broken into its vertical and horizontal components and their intersection is graphed.
2. Press button #2. The vertical component's length is calculated as the difference of the y-coordinates.
3. Press button #3. The horizontal component's length is calculated as the difference of the x-coordinates.
4. Press button #4. The Pythagorean Theorem is applied to this right triangle. The square root of this equation yields the hypotenuse length, AB.
5. One at a time, drag points A and B. All of the *dynamic* coordinates & calculations adjust for the movement.
6. If you want points A and B to have integer coordinates only, go to the top pull-down menu "Graph" and release on "Snap Points". When dragged, they will jump to these spots.
7. At any time, press the "Reset" button to restore the screen to its initial state.

"Linear Programming (with examples)"

OPTIMIZE AN EXPRESSION WITHIN A FEASIBLE REGION

1 animation and additional comments can be accessed from the page tabs (lower left of GSP screen):

In this animation, the user chooses constraints, determines the feasible region, chooses an expression to optimize, and then drags a point throughout the feasible region to find where its coordinates cause the optimum value.

How to run this animation: **Directions appear on the main screen. Read them.**
An expanded version of those directions (and sample lesson) appears below.

1. **Set Constraints**: There are a maximum of 7 possible constraints. To set them, one at a time ...
   a) Press (single-click) one of the 7 "Show" buttons (3 at right are for vertical lines only).
   b) Double-click the new constraint's expression to edit it. In the window that opens, enter the border line equation for the constraint in the form of y = mx + b. The " y = " part of the equation is automatic, do not try to enter it.
   c) To indicate one side of the border, drag the two points at either end of the giant arrows. The base endpoint slides the giant arrow along the line. The arrowhead endpoint swivels the arrow around to face whichever direction you want to indicate.

2. Do not "Show" more constraints than are needed in the problem.

3. Along the way, move or rescale the axes as needed (see directions at bottom right of sketch).

4. **Shade Feasible Region**: Once all needed constraints are in place and giant arrows are positioned appropriately, a bounded polygonal region, called the feasible region, is determined. To shade it, you must "construct" the polygon's interior. *First, click in a blank area so no objects are currently selected*. Next, click at any one vertex of the polygon (you won't see a point here at first, but once you click the mouse, a point will appear and it will be highlighted). Now continue clicking at each vertex of the polygon progressing either clockwise or counterclockwise around the perimeter of the polygon until you have selected each of its vertices (do not repeat the first one). You may want to hold down the shift key as you do this (the shift key is needed only if you accidentally click in a blank space so that you don't lose the vertices that already are accumulated.) *With all vertices highlighted*, click on the menu at the top of the GSP screen labeled "Construct". From its options, click on "(polygon) Interior". The shading appears.

5. **Edit "a" and "b" for the expression you wish to optimize**: At the bottom of the GSP screen, double-click on "a =.." or "b =.." to edit these values. The red dynamic measure (ax+by) uses these values along with the coordinates of point P substituted for x and y.

6. **Drag the large red point "P"** (originally it will be near the origin) into the shaded feasible region. Keep dragging it inside this region until you have found the optimum value of the expression edited in step 5. (In this way, students will discover for themselves that the optimum values occur at vertices of the feasible region.) If you wish to hide the constraints, but keep the constructed polygon, press the "Hide Constraints?" button.

7. **To move point P to a specific ordered pair**: (more precise than dragging) Click on the button labeled "Show Steps" in the yellow directions box. Follow the directions that appear at the upper right of the GSP screen. This allows you to type in the coordinates for your targeted location and the "Move" button will move P to that exact spot.

** After finishing one problem and before starting another one, you may wish simply to reopen a fresh copy from your saved file, or use "Edit" (+ hold shift key) -> "Undo All", or press all "Hide" buttons and manually delete the polygon you constructed.